



The Handbook

Codex

[http://codex.wordpress.org/CSS Troubleshooting](http://codex.wordpress.org/CSS_Troubleshooting)

Version Date

6 August 2005

CSS Troubleshooting

With the introduction of the new [Themes](#) in [WordPress v1.5](#), boring and commonplace website layouts have become a thing of the past. With a few clicks, you can change your layout instantly. With a few more clicks and tweaks, you can screw up your layout instantly as well. Welcome to the exciting world of web page design.

When you encounter a screw-up in your layout, many people come running to the [WordPress Forums](http://www.wordpress.org/support/) (<http://www.wordpress.org/support/>). While the willing volunteers can do what they can to help you, there are some steps you can take to get to the solution, or at least a better idea of where the problem may lie, before you get to the Forums.

[Know Before You Go](#)

We have a list of things you need to know before you go to the forums with layout design problems, and tips on solving the problems yourself.

[Examine Your HTML and CSS](#)

Take a close comparative look at your HTML and CSS and make sure that all the references match.

[Isolate Your CSS Challenges](#)

Below we'll show you a couple of techniques to help identify the areas that are causing your problems in an effort to narrow down the problem to a specific area and code.

[Common CSS Errors](#)

You are not the first to have this problem. We have a list of some of the most common CSS errors to help you fix the little details that can mess up your layout.

[Pest Control - Watching Out For Browser Bugs](#)

While we will help you identify some of your CSS challenges, a lot of them come from bugs and conflicts between browsers, so we'll give you some tips on how to work around the various browser bugs.

It is the goal of this article to help you solve your layout design problems within the CSS file, not within the HTML or PHP files. For help on modifying those, check out [Using Themes](#) for more information.

BACKUP

Before beginning any of these problem-solving tips and techniques, be sure and [backup your data](#) just in case. Also, backup the files you are working on as you try different things so you have some places to go back along the way.

Know Before You Go

If you are new to CSS and web page design, start with a visit to [WordPress' CSS Tips, Techniques and Resources](#) to find information on the basics of CSS and possibly answer some of your questions. At the least, you will get a basic overview of what CSS is, the

impact it has on the HTML or structure of your page, and learn some jargon to help you ask a more informed question on the forums.

You will also need to know some basic terminology to help you express your problem to others. This isn't a how-to-CSS guide but more of a "what is thingamahjig called" guide.

CSS (Cascading Style Sheets) are bits of code that influence the presentation or the look of your page's HTML code. In WordPress, the CSS styles are generally found in a file called `style.css` in the specific [Theme](#) folder you are using. The HTML code and CSS references that hold the structure of your page are generally found in the `index.php` file in your Theme folder.

The PHP files are found within your Theme folder and contain the code and references which generate your HTML page. It is here, in the final run, that you may make changes to the specific CSS selector tags, not your HTML page. For help on modifying those, check out [Using Themes](#) for more information.

'CSS *selectors* (names) are generally grouped into three specific references: The **ID**, **CLASS**, and HTML tags.

The ID

The **ID** is a reference to a specific unique area on your web page. It is generally seen represented on your HTML web page as an enclosed DIV (division) block:

```
<div id="header">Title of the Page</div>
```

In the style sheet (CSS) the **ID** selector is referenced as `#header` and might look like this:

```
#header { position: relative; margin:0; padding:0;
         height:100px; width: 100%; background: red;
         color: white;}
```

The CLASS

The **CLASS** is a reference to any element on a page that needs to look a specific way when this reference is used. For example, if you frequently want to highlight a word or two within your text (we'll use red as a highlight color in this instance), you might have a **CLASS** selector in your style sheet like this:

```
.hilite { color: red}
```

and the reference in your HTML might look like this:

```
...this is some text about something
I want <span class="hilite">in red</span>. And
some more rambling here...
```

As you can see, the difference between **ID** and **CLASS** selectors in the style sheet is that an **ID** uses a pound sign (`#name`) and a **CLASS** uses a period (`.name`). **ID** references must be unique on a page and used once. **CLASS** references can be used repeatedly in the same page.

HTMLTags

If you want to "design" a specific HTML tag reference, such as a `blockquote`, the code within the web page may look like this:

```
<blockquote>This is a pithy and brilliant quote  
that I knew you would enjoy.</blockquote>
```

In the style sheet, the reference to the `blockquote` would not have a # or period but would just simply list the HTML and then the design elements. This example indents the quote on both sides and puts a blue line on the left side of the quote and makes the text italic.

```
blockquote { position: relative; margin: 10px 50px 10px 50px;  
padding: 5px; font-style: italic;  
border-left: solid blue 4px; }
```

You can apply any design elements to any specific HTML tag such as `body`, `p`, `H1`, `H2`, `H3`, `ul`, `li`, and more.

For information on modifying **ID**, **CLASS**, and HTML, check out the resources at [WordPress' CSS Tips, Techniques and Resources](#).

Important note! Certain themes have their css styles in more than once place, the Kubrick theme being one of them. If, after changing css attributes in a css file, you don't see the results you wish, it's possible your theme choice has some embedded css in one of the php files (usually a header file) and the embedded css styles are overriding the linked or imported css stylesheet.

Examine Your HTML and CSS

In order to determine what is going right or wrong with your web page layout, you have to go to the source. This means looking under the hood.

Under the pretty hood of your web page, the nice layout you see on your Internet browser, is a whole bunch of code with strange and foreign sounding references. At first glance, it's like looking under the hood of a racing car. You know that all of that junk makes the car go, but what the heck is all that mess under the hood?

Viewing HTML

To lift the hood on your page, make sure the page is being viewed from your web browser and from within your web browser's top menu, click VIEW > SOURCE or PAGE SOURCE. Another page will pop-up either inside another browser window or inside of a program that comes with your operating system called Notepad, or some variation thereof. This is your HTML page which structures your page.

Viewing CSS

To view your CSS, either know the actual address (or have it already on your hard drive), or scroll down through the HTML page to the following reference:

```
<link rel="style sheet" type="text/css" href="wordpress/wp-
```

content/themes/default/style.css">

This is the link that loads in your attached CSS style sheet. To view your CSS, either double click on the file name or type in the specific link to the file in your web browser such as:

<http://www.yoursite.com/wordpress/wp-content/themes/default/style.css>

In WordPress, PHP is used to actually generate the HTML page. This is often complex and confusing code. To view the HTML, view a generated page, such as a sample post. To make changes in the HTML structure and CSS references, you will need to modify the appropriate PHP file. [The Codex page on using themes](#) has more information on how to view your Theme Templates and find out which Template is associated with which section on your page.

The problem-solving techniques herein describe how to change the CSS to influence the layout of the page. To make actual changes to the Themes, check out [Theme Development](#).

Looking Under the Hood for CSS

Once you have lifted the hood to see the HTML and CSS, it is time to start playing Sherlock Holmes. To start, you have to know what you are looking at and where to find the basic elements to help you find the culprit.

The HTML page features all the structural code that sets up the "grid" into which your page sits. Think of it as a map with notes written all over it. The notes are actually pointers to the directions which are found within the CSS file.

Scroll down past a lot of `<link rel='archives' . . .` information until you see `<body>`. Below this will be the layout "body" of your web page. From here on, every bit of information is critical to the structure and design of your page.

As you scroll through it, look for identifying ID and **CLASS** statements. For example, you may see the following:

```
<div id="page">
  <div id="header">
    <h1>My WordPress Site</h1>
    <div id="headerimg">
      <div class="description">by Me and Only Me</div>
    </div>
  <div id="content" class="widecolumn">
    <div class="post">
```

This may look confusing, but it is basically setting up sections with your content in it.

It begins with the division called "page" which sets up the look of the entire page. If you look in the CSS file for `#page` selector, you will see the presentation styles associated with it. It is followed by the *header* division which includes a heading (`H1`) with the title of your site. This particular layout is based upon the default [Theme for WordPress v1.5 called Kubrick](#) which includes an image in the header, set by the *"headerimg"* division. After that comes a division with a **CLASS** reference called *"description"* which is the place you have a subtitle or description of your site. Again, look in your CSS to find `.description` to find out how that area is styled.

The next two tags close the *header* (`</div>`) and then begins the division ID of *"content"* which also features a **CLASS** called *"widecolumn"*, followed by another **CLASS** called *"post"*. The *"content"* and *"widecolumn"* layout the overall look of the containers holding your *"post"*.

This is an example of the parent/child relationship. And, as you will see in the next section, this is one of the major places where CSS layout problems begin!

The CSS Parent and Child Relationship

One of the biggest problems in designing web pages is understanding where a problem occurs and whose influence might be affecting the problem. This is called the "parent/child relationship" of CSS. As you know, while parents usually have their children's best intentions at heart, children often feel intimidated and screwed up by their parents, so understanding this relationship may help you solve your problems.

A WordPress user posted a question on the forum complaining that she wanted her page to feature her *header* spreading fully across the page's width with the content centered on the page with a lot of space on the left and right sides. She's been poking around with the margins in her *header* to no avail and turned to us for help.

```
<div id="page">
  <div id="header">Header Title</div>
  <div id="content">
    <div class="post">Post babble here...</div>
```

The CSS attributes for this margins in this example are:

```
#page { margin-top:5px; margin-right: 100px;
        margin-bottom: 5px; margin-left:100px; }
#header { margin-top:5px; margin-right: 5px;
          margin-bottom: 5px; margin-left:5px;}
#content { margin-top:5px; margin-right: 20px;
           margin-bottom: 5px; margin-left:20px; }
.post { margin-top:5px; margin-right: 5px;
        margin-bottom: 5px; margin-left:5px; }
```

Playing detective with her codes, we found that changing the margins on the *header* weren't working because they were being influenced by the *page* margins to begin with. This is where the parent/child relationship shows up. The parent *page* was telling the child *header* what to do and it wanted to do something else.

If we changed the right and left margins of *page*, it eliminated the margin problems for the *header*. But we've created another problem. The parent *page* continues its influence and now all of the content is spread across the whole page width. A change needed to also happen with the left and right margins of the *content* so the wide margins are back in place. To make the whole family happy, the new margins looked like this:

```
#page { margin-top:5px; margin-right: 5px;
        margin-bottom: 5px; margin-left:5px; }
#header { margin-top:5px; margin-right: 5px;
          margin-bottom: 5px; margin-left:5px;}
#content { margin-top:5px; margin-right: 100px;
           margin-bottom: 5px; margin-left:100px; }
.post { margin-top:5px; margin-right: 5px;
        margin-bottom: 5px; margin-left:5px; }
```

Isolate Your CSS Challenges

Identifying the parent/child relation influences over your web page layout helps to solve a lot of problems, but sometimes the relationships are so complex, it's hard to figure out which section is which and who has influence over which parts.

To isolate and identify your various CSS sections, divisions, and classes, here are some simple tricks. Before you begin, be sure and **make a backup** of all of your main files including your CSS to make sure you have something to recover from if this gets out of control.

Once you have identified the culprit and fixed it, make sure you remove these testing features so your web page will look "normal" again.

Box Your Sections

Once you have identified the various sections or divisions within your HTML page, go into your CSS file and add the following attribute to the various divisions:

```
border:solid 1px color
```

Where the word *color* is, put a different color name in place for each section.

For example:

```
#page { margin:5px; padding:0; border: solid 1px red; }
#header { margin: 10px; padding: 5px; border: solid 1px blue; }
#content { margin:5px 100px 5px100px; padding:10px;
           border: solid 1px green; }
.post { margin:5px; padding:10px; border: solid 1px yellow; }
```

Save the CSS file and view your page (click REFRESH or F5) in the web browser. You should now see a different colored box around each of the different sections:

This is a section of rambling text that goes on and on.

This is another section that has been highlighted in a red box.

This is the rest of the text back to normal.

If you don't see a colored box around your content, check again that the selector you changed is actually the correct spelling and identified in the HTML.

If the problem you are having is in the blue box, then you know where to start solving your problems. Be sure and remove the test attributes when you're done.

If you use Firefox as your browser there's a handy extension that does this for you, and makes troubleshooting css problems a whole lot easier. The Plugin is called Aardvark and is available for free at <http://www.karmatics.com/aardvark/>. A demo is available for test drives.

Highlight Sections

Besides putting boxes around the different sections to isolate the problem CSS or HTML, you can dramatically change the colors of the content to make the problem "jump" right out at you. By changing the text color or background color of a section, you will spot it immediately when you view the screen. **Note:** *Be sure and make note of the original colors if you change them during testing so you can go back to them. And make frequent backups!*

In the CSS file, you can change a section's font color by adding `color:red` or any other color to the selector's attributes such as:

```
H1 { font-style:bold; font-size: 125%; color: red; }
```

The H1 heading should jump out at you in bright red and would look like this:

To change the background color of a section, you can add `background:pink` to make the entire background pink.

```
#header { margin:5px; padding: 10px; background:pink; }
```

The result might look like this:

This is some text that goes on babbling here and there.

This is some text with the background color changed so you can see it.

This is the rest of the text back to normal.

The entire header division will now feature a pink background. When you've identified the culprit, and made the fixes, be sure and remove any testing attributes to restore the look of your web page.

Validate Your Source Code

Sometimes the smallest detail can send your page out of whack. A misspelled tag (`rhref` instead of `href`), a forgotten closing tag, a missing attribute, or even the wrong attribute can send your page into a design tail spin.

Free online validators which check your HTML, XML, and CSS code may help isolate the little detail you are missing. As you scan through the code, it's easy to skip over a little stumble. Most online validators let you either type in the URI (link) to your site to initiate the validation process, or may even allow you to paste in code or upload a file to have it inspected. WordPress, by default, validates its default coding, but if you are making modifications, the slightest slip can screw things up.

Different validators check for different problems, so if you can't find your solution with one validator, try another. Many validators will even recommend making some changes, but find your problem before you start creating new ones.

To help you understand more about validation and to find online validators, we've provided a list of validation resources in a Codex article called [Validating a Website](#).

Slash and Burn - The Last Resort

Not for the timid

NOTE: There are two Slash and Burn techniques. If you are weak of heart, check out the [second one](#).

If you can't seem to narrow down the problem, there is a technique, sometimes called "Slash and Burn", that will help you narrow down the culprit. **It requires no interruptions, concentration and thorough backups to ensure you don't destroy even the screwed up remains of your web page design.** We also recommend you have familiarity with HTML and CSS.

1. Make backups of all of your files.
2. Open a post in the web browser and VIEW > SOURCE.
3. Save this source file as a text document called "junk.html" to an empty test folder on your hard drive. DO NOT CLOSE THIS FILE. It will remain open during this entire process.
4. Copy your CSS file to the same test folder.
5. If you are having problems with the graphics, copy the graphics folder or the graphics to the test folder.
6. In the junk.html source file, change the style sheet reference from something like

```
<link rel="style sheet" type="text/css" href="wordpress/wp-content/themes/default/style.css">
```

to this:

```
<link rel="style sheet" type="text/css" href="style.css">
```

Save the junk.html text file (DO NOT CLOSE IT).

7. In the test folder, double click on junk.html to view the file in your browser. You should see the general layout of the page with the graphics, if appropriate. If not, double check the link reference to the style sheet.
8. In your junk.html text file, move to the point where the trouble begins. Move to the section above (a section which includes opening and closing tags such as:

```
<p>babble...</p>
```

or

```
<div class="post">babble...</div>
```

and highlight the entire section from the opening tag to the closing tag and CUT the section (Cntrl+X).

9. Save the file.
10. REFRESH the web page in the web browser (F5 or click REFRESH - if you have problems and don't see a change, hold the SHIFT key then press F5 or (in FireFox) simultaneously hold down Cntrl+Shift+R).
11. You should see the removed section missing. Check below to see if this fixed the problem or if it went away. If yes, this section is your problem. If not, go to the next step.
12. If the problem is still there, move back to the junk.html file and put the cursor in the place where you deleted the section, if the cursor has moved. PASTE the cut section back in (Cntrl+V). Move to another section above or below this point and repeat steps 8 through 12.

At some point in this process, you will see the problem either fix itself or disappear. Begin with large sections and when you find the large section problem area, break it up into smaller pieces. Eventually, you will isolate the area that is causing you grief. Note the CSS references to identify the troublesome section and start making changes to the CSS file to fix it.

Gentle Slash and Burn

To use the *gentle* version of slash and burn, instead of deleting the sections as shown above, cut and paste them into Notepad or another text editor so they are protected in case you get distracted from the cut and paste process. ALWAYS back everything up as you go along, *just in case* (which happens a lot more than you might think!).

Common Errors

We all make mistakes. The word "typo" wasn't invented without reason. Here are some of the most common problems that creep up with CSS.

Missed Spellings

Just so you know, `leftt` is not the same as `left` and this could be the reason something is on the right instead of the left side of your page. Putting a `30ps` for a margin won't get much of a result, but `30px` will. Missed spelling errors are common and easy to overlook. Luckily, CSS validators can often catch these tiny mistakes for us.

Forgotten Details

As creative as you can be with CSS, there are some ground rules you have to follow. Every selector must be identified as an ID or CLASS unless it is a HTML TAG. It must be laid out as `selector { property: value; property: value; }` and the braces, colon and semi-colon must be there. Miss one of these little details and nothing will happen, or strange things might. CSS validators will usually catch these little forgotten details for you.

Wrong Selector

Putting all your wonderful designs in `#content` when they were supposed to be in `#context-text` doesn't help your layout. Luckily, you can usually see these immediately upon viewing the page, so just cut and paste them in the right tag...and then remember what you deleted from `#content`. Of course, you can refer to your frequently backed-up file to get the lost code. Hint-Hint!

Wrong Template Module

As useful as WordPress' new modular templates are, many a time a user has made a modification in `comments.php` instead of `comments-popup.php` or other similarly named files by mistake. Double check which modular section you are *supposed* to be working on all the time. And if you mess one up by accident, there is always that faithful backup file to make things new again.

Multiple Choice

CSS can't read your mind. If there are two references to the same selector with conflicting information, it has to decide which one it will use. This is very common if you are bringing your original style sheet in on top of a new one. If you are fighting with a selector for, say, the `h1` heading, and nothing is changing, search through the style sheet to see if there is another reference to that selector.

Pest Control - Watching Out For Browser Bugs

No matter how hard we try to make our web pages beautiful, it can take the viewing of the web page in a different browser to totally screw up our lovely design.

Different browsers view web pages differently. Older browsers won't recognize new CSS standards, while newer ones often "see" things differently from their brethren. The results are often not pretty, causing blinking, jumping, or missing design elements, shifting layouts, and distorted positions.

How do you know it's the browser and not your design causing the problem? You often don't. If you play with CSS a lot, you will learn to recognize the symptoms. In general, here are the clues:

- Text jumps around or blinks as you scroll down the page.
- Links and text jump around as you move the mouse over a link.
- Lists with links jump around and move as you move your mouse over them.
- Layouts look different. Columns are wider or narrower between browsers.
- Graphics overlap the text or lists.
- Special effects such as filters don't look the same or aren't there.

To get help with your browser bugs, check out the recommendations and resources at [CSS Fixing Browser Bugs](#).